Technical Reports on Information and
Computer Science from Kochi
Vol. 1 (2009), No. 5

FDL 処理系の移植と今後の開発方針について

About the transplant of the FDL converter and its development policy in the future

多和田　侑　　森　雄一郎
Yuu Tawata　　Yuichiro Mori

高知大学　理学部数理情報科学科
Dept. of Mathematics and Information Science, Kochi UNIV.

概要

ファジィ理論を利用したシステム（ファジィシステム）の開発用言語として，ファジィシステム記述言語（FDL：Fuzzy systems Description Language）が策定された．同時に，FDL 試作処理系（fdltoc）が開発された．しかし，OS 依存性が高く，OS によっては動作が不安定であった．そこで，本研究では fdltoc を一般的な OS の UNIX 環境への移植を行い，動作を確認した．そこで，FDL 処理系を公開し，処理系に対する意見や，FDL 仕様に対する意見を頂き，今後の FDL の開発方針の参考にしたいと考えている．本論文では，FDL の開発方針の 1 つとして組み込み系ハードウェアに展開させることについて述べる．

1　はじめに

現在，ファジィ理論を応用したシステムを搭載した製品開発や研究が様々な研究機関で行われている．それらのシステムで利用しているファジィ推論のための知識ベースは，実行環境の違いはあるが，記述方法や構成は似通っている．だが，実装する言語，ハードウェアが異なるために，システムを他の環境へ移植するのが難しい．システムの比較・検討が容易でない．システム開発に関係のない人はプログラムをみても理解が難しいなどの問題が起きていた．その上，蓄積したデータを広く共有できないというソフトウェア資源の問題も生じてくる．

これらの問題を解決するために，ファジィ理論を記述できる共通規格の開発言語が必要となった．統一された言語を利用することにより，システム開発の円滑化が期待でき，知識データを広く共有することも容易になると考えられ，業界の健全な発展が望める．

2　ファジィシステム記述言語

2.1　提案

ファジィシステム記述言語（以下，FDL）はファジィ理論を記述できる言語として策定された．FDL の仕様策定はいくつかの公的機関によって進められ，1996 年に当時の（社）日本電子機械工業会（EIAJ）によって仕様が決定された．

2.2　仕様

FDL の標準仕様は 1996 年に決定された． FDL はファジィ理論を用いたシステム全般の開発を目的とした言語である．従って，FDL の対象領域は極めて広くなり，ファジィデータの記述，演算，推論が容易でなければならない．また，プログラミング言語としての基本機能も備えなければならない．だが，対象領域が大きい，ファジィ理論自体に未解決な部分が存在することから，当初は FDL の対象領域をファジィ制御の分野に限定して開発された．

現在の仕様としては必要最低限なフ

ァジィ推論を行うために，主に，以下の表にあるような機能を備えている．

ファジィデータを扱う型	ファジィ真理値型 ファジィ型
ファジィ値表記	列挙表記 シングルトン表記 関数表記 補間列挙表記 ベクトル表記 三角形表記 台形表記
ルールベース	rule_base 関数
演算子	and or not

表 1　拡張された機能例

※演算子（and，or，not）の機能を選択することができるため，様々なファジィ演算に対応することができる

また，FDL はシステム全般の開発を目的としているため，システム開発用言語として最も普及している C 言語（ANSI-C）をベース言語として採用している．

3　FDL 処理系

3.1　処理系（fdltoc）について

FDL は C 言語をベースとした高級言語である．すなわち，FDL プログラムのコンパイル方法には 2 つの方法がある．1 つめの方法は，ソースコードをハードウェア固有の機械コードに直接変換する方法である．2 つめの方法は既存のコンパイラを利用する方法である．つまり，コンパイラが扱うことのできる言語に変換した後に，コンパイラを利用して機械コードに変換するという方法である．

1 つめの方法では特定のハードウェアに対してのみ有効な処理系（コンパイラ）ができてしまい，ハードウェア毎にコンパイラを作成しなければならない．従って，2 つめの方法を用いて既存のコンパイラを利用することで，様々なハードウェアに対応できるような処理系を作成している．

処理系（fdltoc）は 1995 年に明治大学の向殿・石畑研究室のメンバーによって作成された．fdltoc は GNU Bison/Flex を利用した言語解析・変換プログラムであり，FDL のソースを C 言語のソースへ変換することを目的としている．

以上のことからわかるように，fdltoc はコンパイラではなくコンバータである．C 言語のソースへと変換後に ANSI-C コンパイラを用いてコンパイルし，実行ファイルを作成する．FDL は C 言語を拡張した言語であり，C 言語の上位互換言語である．すなわち，FDL プログラム内に C 言語の記述を含めることができる．

既存の言語に変換する利点は大きく，プログラムの可読性を高めて，今あるソフトウェア資源を活用することもできる．変換する言語を C 言語とした理由は，FDL が C 言語ベースとした言語であるために，その変換が容易であること，C 言語のプログラマ習得率の高いこと，システム開発用に構築された C 言語がハードに最も近い高級言語であることを考慮したためである．

処理系は現仕様の一部の仕様の実装を見送っている（シングルトン表記，関数表記，ラベル機能）．

3.2　処理の流れ

FDL のソースから実行ファイルを作成するまでの処理の流れについて述べる．簡潔に説明するならば，fdltoc を用いて変換後に C コンパイラ（以下，gcc）を用いるのである．そうすれば実行ファイルを作成することができる．

gcc は 4 つのプロセス（プリプロセス，コンパイル，アセンブル，リンク）を経て実行ファイルを作成する．それに対して，厳密な fdltoc の実行位置はプリプロセスとコンパイルの間となる．これは，プリプロセスの役割を fdltoc に含めくめないためであり，fdltoc の役割を最小限に抑えることにつながる．また，コンパ

イル後にしてしまうと，拡張された FDL の文法が必ずエラーとなってしまう．以上のことより，fdltoc の実行位置はプリプロセスとコンパイルの間に行うのが最適だと考えられる．よって，次の図のような関係になる．

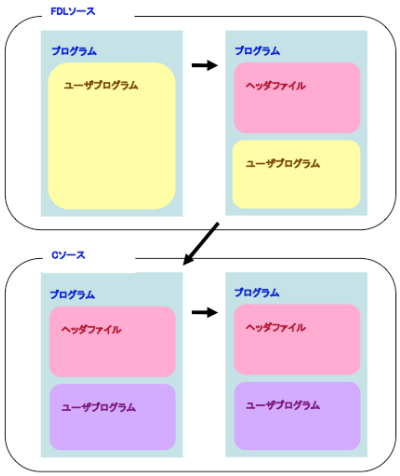

図 2　処理の流れ

順に，プリプロセス，fdltoc、コンパイラを実行した後のソース内部の構成を示した者である．図を見てわかるように，fdltoc はユーザプログラムと C 言語のヘッダファイルを解析しなければならない．

4　fdltoc の移植

4.1　移植の必要性

FDL の処理系として fdltoc が作成された．しかし，fdltoc は開発された環境である SunOS 以外での OS 上では動作が不安定であった．FDL は様々な環境上で動作することが期待される．従って，fdltoc の移植を行う必要がでてきた．

4.2　移植する上での問題点

移植する上で問題点となったのは，C 言語処理系の OS 依存性である．

OS 依存性は簡単なプログラミングを行う上で考慮することはほとんどない．つまり，プログラマが記述する部分にはほとんど OS 依存性は存在しない．OS 依存性はヘッダファイルに存在する．ヘッダファイルには OS と情報をやり取りするために C 言語の本質にあたる定義が記述されている．実際に，OS によって処理系のヘッダファイルには異なる記述が多く存在する．それは，処理系が各 OS のアーキテクチャに少なからずとも依存せざるを得ないからである．

また，OS 依存関数というのも存在する．それは，特定の OS 上でしかサポートされておらず，他の OS 上では関数名が異なる，関数自体が存在しない，ということもあり得る．よって，OS 依存関数を使用していては他の環境では動作しないという可能性があるため，OS 依存関数についても考慮しなければならない．

4.3　処理系の改善

問題となっていたのは C 言語処理系の OS 依存性である（OS 依存関数に関してはただひとつの関数を置き換えただけであるので，詳細は省く）．

OS ごとに違うヘッダファイルに対応させていては既存の処理系を利用する利点を損ないかねない．よって，この問題を解決するために，字句解析の時点でヘッダファイルに関する記述を読み飛ばすこととした．即ち，解析・変換を行うのは，ユーザが作成したプログラム本体のみである．こうした仕様にした理由は，FDL はファジィ理論に関する記述を C 言語から拡張したものであるので，解析・変換を行うのは，拡張された記述だけでよいからである．

4.4　動作確認

動作確認を行ったのは，MacOSX（Darwin），Windows（cygwin），Linux（Ubuntu，Vine）である．これらの UNIX 環境上での正しい動作を確認できた．

5　FDL の今後の開発方針

FDL の開発方針としては，1 つ目に他

の環境に対応した FDL 仕様策定, 処理系開発. 2 つ目に FDL プログラムの開発を支援する統合開発環境の開発. また, fdltoc の改善, FDL 現仕様の拡張などが考えられる. FDL 現仕様はファジィ制御を行うための機能だけに特化した仕様となっているため, 研究用としては機能が少ない. 重要な仕様拡張として, 数値処理系から数式処理系への拡張があげられる.

このように FDL の開発方針は様々なものが考えられるが, 本研究で次の開発方針としてあげるのは, 組み込み系ハードウェアに対応した FDL 仕様の策定, 処理系開発である.

6 組み込み系ハードウェア

ファジィ理論は利用した制御法をファジィ制御という. ファジィ制御は主に組み込み系ハードウェアに取り入れられており, その理由としては, 人の感覚に対応することができる, 少ない情報で推論することができるからである. つまり, 組み込み系に FDL を対応させることで多くの開発現場で FDL を普及させることができると考えられる.

だが, 組み込み系プログラミングは通常のプログラミングと異なる点が多くある. それは, メモリ領域定義, 初期化処理である. また, 計算精度がハードウェアに大きく依存し, ライブラリもまた統一されたものが存在するわけではない. 組み込み系に対応した FDL 仕様策定をする上で, これらの問題をどう解決するかが重要である.

7 まとめ

FDL 処理系（fdltoc）を一般的な OS 上の UNIX 環境で動作するように移植作業を行った. これにより一般的な PC 上で FDL プログラミングが可能になった. よって, fdltoc を試験運用していただくために森研究室の HP

（http://www/is/kochi-u.ac.jp/~ymori/fdl/）

で配布している. FDL に関するご意見をいただき, 今後の開発の参考にしようと考えている.

謝辞

本論文は, 著者が高知大学理学部数理情報科学科 4 回生在学中に森研究室にて行った卒業研究をまとめたものである.

本研究の全過程を通じて収支親切丁寧な直々のご指導をして下さいました森雄一郎准教授に心より感謝致します.

また, 本論文執筆中に支えてくださった森研究室の皆様に深く感謝致します.

参考文献

[1] "ファジィシステム記述言語（FDL）標準仕様解説書",（社）日本電子機械工業会（EIAJ）編, 1996

[2] 鑓田威, 石畑清, "ファジィシステム記述言語の試作処理系", 11th Fuzzy System Symposium, p23-26（1995）

[3] 石畑清, 森雄一郎, 大塚和彦, 新沢洋太, 向殿政男, "ファジィシステム記述言語の標準化に向けて", 11th Fuzzy Systems Symposium, p27-30（1995）

[4] 大塚和彦, 森雄一郎, 向殿政男, "ファジィシステム記述言語の標準化に向けてⅡ", 14th Fuzzy System Symposium, p291-292（1998）

[5] 草木原貴郎, "FDL の標準化に向けての提案", 高知大学大学院理学研究科情報科学学位論文

[6] 鹿取祐二, "C 言語で H8 マイコンを使いこなす", オーム社, 2003

[7] 多和田侑, 辻由彦, 森雄一郎, "FDL の現状と今後の方向性", 平成 20 年度日本知能情報ファジィ学会中国・四国支部講演論文集, p49-50

[8] 多和田侑, 辻由彦, 森雄一郎, "FDL 処理系の移植と今後の開発方針について", 平成 20 年度電気関係学会四国支部連合大会講演論文集, p326